IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Patent Application

Applicant(s):	Robert A. Corley
Docket No.:	2
Serial No.:	10/675,718
Filing Date:	September 30, 2003
Group:	2616
Examiner:	Abdulla A. Riyami
Title:	Processor with Continuity Check Cache

APPEAL BRIEF

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir:

Applicant (hereinafter "Appellant") hereby appeals the final rejection dated March 18, 2008 of claims 1-20 of the above-identified application.

REAL PARTY IN INTEREST

The present application is assigned of record to Agere Systems Inc. On April 2, 2007, the assignee Agere Systems Inc. completed a merger with LSI Logic Corporation, with the resulting entity being named LSI Corporation. LSI Corporation is the real party in interest.

RELATED APPEALS AND INTERFERENCES

There are no known related appeals or interferences.

STATUS OF CLAIMS

The present application was filed on September 30, 2003 with claims 1-20, all of which remain pending. Claims 1, 19 and 20 are the pending independent claims.

Each of claims 1-20 stands rejected under 35 U.S.C. §103(a). Claims 1-20 are appealed.

STATUS OF AMENDMENTS

There have been no amendments filed subsequent to the final rejection.

SUMMARY OF CLAIMED SUBJECT MATTER

Independent claim 1 is directed to a processor. The processor comprises controller circuitry operative to control performance of a continuity check for each of a plurality of flows of protocol data units received by the processor and memory circuitry comprising a continuity check cache. The continuity check cache stores an identifier for each of a subset of the plurality of flows. The controller circuitry controls access to a set of continuity check counters comprising a counter for each of the plurality of flows.

The controller circuitry determines if a given flow for which a protocol data unit is received in the processor has a corresponding entry in the continuity check cache. If the given flow has such an entry, the controller circuitry prevents a corresponding one of the continuity check counters from being updated. If the given flow does not have such an entry, the controller circuitry clears the corresponding one of the continuity check counters and stores a flow identifier for the given flow in the continuity check cache.

In illustrative embodiments described in the specification at, for example, page 5, lines 6-14, with reference to FIG. 1, and page 12, lines 4-15, with reference to FIG. 4, a processor (e.g., 102) comprises controller circuitry (e.g., 120) operative to control performance of a continuity check for each of a plurality of flows of protocol data units received by the processor and memory circuitry (e.g., 104) comprising a continuity check cache (e.g., 122). The controller circuitry controls access to a set of continuity check counters (e.g., 124 in external memory 106) comprising a counter (e.g., counters 124_0, 124_1, . . . 124_N) for each of the plurality of flows. As described in the specification at, for example, page 7, line 19, to page 8, line 1, with reference to

FIG. 2, the continuity check cache stores an identifier (e.g., 200) for each of a subset of the plurality of flows (e.g., flow 1, flow 2, . . . flow M).

As described in the specification at, for example, page 9, lines 17-21, with reference to FIG. 3, the controller circuitry determines (step 304) if a given flow for which a protocol data unit is received (step 302) in the processor has a corresponding entry in the continuity check cache. As described in the specification at, for example, page 9, lines 22-27, with reference to step 308 in FIG. 3, if the given flow has such an entry, the controller circuitry prevents a corresponding one of the continuity check counters from being updated. As described in the specification at, for example, page 10, lines 1-5, with reference to step 306 in FIG. 3, if the given flow does not have such an entry, the controller circuitry clears the corresponding one of the continuity check counters and stores a flow identifier for the given flow in the continuity check cache.

Independent claim 19 is directed to a method for use in a processor comprising controller circuitry operative to control performance of a continuity check for each of a plurality of flows of protocol data units received by the processor, the controller circuitry being further operative to control access to a set of continuity check counters comprising a counter for each of the plurality of flows.

The method comprises the steps of storing an identifier for each of a subset of the plurality of flows in a continuity check cache, and determining if a given flow for which a protocol data unit is received in the processor has a corresponding entry in the continuity check cache. If the given flow has such an entry, a corresponding one of the continuity check counters is prevented from being updated, and if the given flow does not have such an entry, the corresponding one of the continuity check counters is cleared and a flow identifier for the given flow is stored in the continuity check cache.

Illustrative embodiments of the method recited in claim 19 may be implementing using a processor as described in the specification at, for example, page 5, lines 6-14, with reference to FIG. 1, and page 12, lines 4-15, with reference to FIG. 4. A processor (e.g., 102) comprises controller circuitry (e.g., 120) operative to control performance of a continuity check for each of a plurality of flows of protocol data units received by the processor. The controller circuitry controls access to a set of continuity check counters (e.g., 124 in external memory 106) comprising a counter (e.g., counters 124_0, 124_1, . . . 124_N) for each of the plurality of flows.

As described in the specification at, for example, page 7, line 19, to page 8, line 1, with reference to FIG. 2, an identifier (e.g., 200) for each of a subset of the plurality of flows (e.g., flow 1, flow 2, . . . flow M) is stored in a continuity check cache (e.g., 122). As described in the specification at, for example, page 9, lines 17-21, with reference to FIG. 3, it is determined (step 304) if a given flow for which a protocol data unit is received (step 302) in the processor has a corresponding entry in the continuity check cache. As described in the specification at, for example, page 9, lines 22-27, with reference to step 308 in FIG. 3, if the given flow has such an entry, a corresponding one of the continuity check counters is prevented from being updated. As described in the specification at, for example, page 10, lines 1-5, with reference to step 306 in FIG. 3, if the given flow does not have such an entry, the corresponding one of the continuity check counters is cleared and a flow identifier for the given flow is stored in the continuity check cache.

Independent claim 20 is directed to an article of manufacture comprising a computer readable medium having at least one computer program encoded therein. The at least one computer program is for use in a processor comprising controller circuitry operative to control performance of a continuity check for each of a plurality of flows of protocol data units received by the processor. The controller circuitry is further operative to control access to a set of continuity check counters comprising a counter for each of the plurality of flows

The at least one program when executed in the processor implements the steps of storing an identifier for each of a subset of the plurality of flows in a continuity check cache, and determining if a given flow for which a protocol data unit is received in the processor has a corresponding entry in the continuity check cache. If the given flow has such an entry, a corresponding one of the continuity check counters is prevented from being updated, and if the given flow does not have such an entry, the corresponding one of the continuity check counters is cleared and a flow identifier for the given flow is stored in the continuity check cache.

An illustrative embodiment of the claimed article of manufacture comprising a computer readable medium having at least one computer program encoded therein is described in the specification at, for example, page 7, lines 14-18. The computer readable medium may be, for example, an internal memory (e.g., 104) or an external memory (e.g., 106).

Illustrative embodiments of the article of manufacture recited in claim 20 may be executed in a processor as described in the specification at, for example, page 5, lines 6-14, with reference to FIG. 1, and page 12, lines 4-15, with reference to FIG. 4. A processor (e.g., 102) comprises controller circuitry (e.g., 120) operative to control performance of a continuity check for each of a plurality of flows of protocol data units received by the processor. The controller circuitry is further operative to control access to a set of continuity check counters (e.g., 124 in external memory 106) comprising a counter (e.g., counters 124_0, 124_1, . . . 124_N) for each of the plurality of flows.

As described in the specification at, for example, page 7, line 19, to page 8, line 1, with reference to FIG. 2, an identifier (e.g., 200) for each of a subset of the plurality of flows (e.g., flow 1, flow 2, . . . flow M) is stored in a continuity check cache (e.g., 122). As described in the specification at, for example, page 9, lines 17-21, with reference to FIG. 3, it is determined (step 304) if a given flow for which a protocol data unit is received (step 302) in the processor has a corresponding entry in the continuity check cache. As described in the specification at, for example, page 9, lines 22-27, with reference to step 308 in FIG. 3, if the given flow has such an entry, a corresponding one of the continuity check counters is prevented from being updated. As described in the specification at, for example, page 10, lines 1-5, with reference to step 306 in FIG. 3, if the given flow does not have such an entry, the corresponding one of the continuity check counters is cleared and a flow identifier for the given flow is stored in the continuity check cache.

The claimed invention provides a number of significant advantages over conventional arrangements. As described in the specification at, for example, page 3, lines 9-16, the techniques of the invention in the illustrative embodiment process received PDUs in a manner that substantially reduces the number of external memory accesses required in conjunction with the performance of continuity checks, while also limiting the amount of on-chip memory resources that are consumed. The illustrative embodiment provides an efficient mechanism for allowing a large number of continuity check counters in external memory to be reset without unnecessarily wasting memory bandwidth or sacrificing on-chip area. As a result, processor throughput performance is considerably improved, without significantly increasing the size, complexity, power consumption or cost of the processor.

GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

Claims 1-20 are rejected under 35 U.S.C. §103(a) as being unpatentable over U.S. Patent No. 5,872,770 (hereinafter "Park").

ARGUMENT

Claims 1-5, 7-10 and 14-20

As described above, claim 1 is directed to a processor comprising controller circuitry operative to control performance of a continuity check for each of a plurality of flows of protocol data units received by the processor and memory circuitry comprising a continuity check cache. The continuity check cache stores an identifier for each of a subset of the plurality of flows. The controller circuitry controls access to a set of continuity check counters comprising a counter for each of the plurality of flows.

The controller circuitry determines if a given flow for which a protocol data unit is received in the processor has a corresponding entry in the continuity check cache. If the given flow has such an entry, the controller circuitry prevents a corresponding one of the continuity check counters from being updated. If the given flow does not have such an entry, the controller circuitry clears the corresponding one of the continuity check counters and stores a flow identifier for the given flow in the continuity check cache.

It is important to note that claim 1 specifies that whereas the set of continuity check counters comprises a counter for each of the plurality of flows, the continuity check cache stores an identifier for each of a subset of the plurality of flows. Thus, it is inherent in claim 1 that the number of flows for which the continuity check cache stores an identifier is less than the number of counters within the set of continuity check counters.

The Examiner argues that the above-noted limitations of claim 1 are met by the arrangement shown in FIG. 3 of Park, except in so far as Park fails to disclose memory circuitry implemented as a continuity check cache. Specifically, the Examiner argues that the recited set of continuity check counters comprising a counter for each of the plurality of flows is met by the continuity clock timer part 32 in FIG. 3 of Park and that the recited memory circuitry storing an identifier for each of a subset of the plurality of flows is disclosed by Park at column 9, lines 1-12. Appellant notes that column 9, lines 5-8, discloses that "continuity check cell/user cell comparator 31 compares the received channel identifiers VPI/VCI with channel identifiers VPI/VCI of active connections from the continuity check channel cell registration part 34" and that, moreover, Park at column 9, lines 57-60, teaches that the "continuity check cell registration part 34 is adapted to store therein . . . channel identifiers of continuity check connections." Accordingly, it appears that the Examiner is arguing that continuity check cell registration part 34 corresponds to the recited memory circuitry comprising a continuity check cache, which stores an identifier for each of a subset of the plurality of flows.

Park teaches at column 9, lines 23-26, that the "continuity clock timer part 32 includes a plurality of timers corresponding respectively to continuity check connections in the continuity check cell channel registration part 34." See also FIG. 3 of Park, which shows continuity check timer part 32 including N timers (Timer-1, Timer-2, ..., Timer-N) and continuity check channel registration part 34 including N continuity checking cell channels (Continuity Checking Cell Channel-1, Continuity Checking Cell Channel-2, ..., Continuity Checking Cell Channel-N).

As previously noted, the claimed arrangement wherein a set of continuity check counters comprises a counter for <u>each</u> of the plurality of flows and a continuity check cache stores an identifier for <u>each of a subset</u> of the plurality of flows. Accordingly, the number of flows for which the continuity check cache stores an identifier is <u>less than</u> the number of counters within the set of continuity check counters. By contrast, Park teaches an arrangement wherein the number of timers stored in continuity clock timer part 32 <u>is equal to</u> the number of continuity check connections in continuity check cell channel registration part 34.

Accordingly, even if one accepts the Examiner's characterization of the teachings of Park and even if Park were to be modified in the manner suggested by the Examiner in the present Office Action at page 5, second paragraph, i.e., by implementing the continuity memory disclosed at column 9, lines 1-12, as fast access continuity cache memory within the same processor, the resulting modification would still fail to reach the limitations of claim 1.

Rather, the result would be an arrangement similar to that discussed in the present specification at, for example, page 9, lines 1-6, in which one bit is set for each active flow to indicate if the counter for that flow has already been cleared during the timeout window. This arrangement prevents multiple accesses to memory for the same flow, but requires that a bit be stored in internal memory for every possible flow, thus resulting in an excessive consumption of on-chip memory resources. Such an arrangement would fail to reach the advantages associated with illustrative embodiments of the claimed invention, such as substantially reducing the required number of external memory accesses while also reducing the amount of memory resources that are consumed. See, for example, the present specification at page 3, lines 9-16, and page 9, lines 7-12.

In the Advisory Action, the Examiner contends that "a [c]ounter has integer units whereas a timer has seconds as its units." Appellant respectfully submits that this contention, if true, would undermine the arguments made by the Examiner on page 3 of the final Office Action, specifically the Examiner's contention that the recited set of continuity check counters comprising a counter for each of the plurality of flows is met by the continuity clock timer part 32 in FIG. 3 of Park.

That said, Appellant respectfully submits that the Examiner's contention is inconsistent with the present specification. For example, the specification at page 8, lines 14-17, describes an illustrative embodiment in which each increment of a counter corresponds to a time window having a duration of approximately 0.5 seconds. Accordingly, the units by which a counter is incremented may correspond to a period of time having a duration measured in seconds.

The Examiner also states in the Advisory Action that "in the claims 1, 19 and 20, it is not disclosed that the continuity cache stores an identifier that is less than the number of counters. If M is zero, then it would be substantially lower than N counters (timers)." This argument appears to be directed to the limitations of dependent claim 6, rather than of independent claims 1, 19 and 20, and will hence be addressed below with regard to dependant claim 6.

In the final Office Action at page 4, first and second paragraphs, the Examiner contends that "it would have been obvious to a person of ordinary skill in the art to implement continuity memory [so as to] also act as fast access cache memory within the same processor. The motivation . . . would have been to have a dedicated continuity check cell processor having faster and more efficient operation, administration, and maintenance cell processing."

Appellant respectfully submits that even if one were to assume for purposes of argument that the Examiner has provided a legally sufficient statement of motivation so as to support a proper *prima facie* case of obviousness, there is nonetheless sufficient evidence of nonobviousness so as to rebut any such *prima facie* case. For example, the fact that others have used a less advantageous technique, rather than performing the modification deemed obvious by the Examiner, suggests both a long-felt need and failure of others.

Independent claims 19 and 20 contain limitations similar to those recited in claim 1 and are thus believed allowable for at least the reasons identified above with regard to independent claim 1.

Dependent claims 2-5, 7-10 and 14-18 are believed allowable for at least the reasons identified above with regard to independent claim 1, from which each depends.

Claim 6

In addition to being allowable by virtue of its dependency from independent claim 1, dependent claim 6 recites separately patentable subject matter. Specifically, dependent claim 6 includes limitations wherein the continuity check cache has a capacity of M entries and the set of continuity check counters includes N continuity check counters, where M is substantially less than N. The Examiner contends that this limitation is suggested by block 32 in FIG. 3. See the final Office Action at page 4, last paragraph.

Appellant respectfully disagrees and submits that FIG. 3 of Park in fact teaches away by instead showing continuity check timer part 32 including N timers (Timer-1, Timer-2, ..., Timer-N) and continuity check channel registration part 34 including N continuity checking cell channels (Continuity Checking Cell Channel-1, Continuity Checking Cell Channel-2, ..., Continuity Checking Cell Channel-N). See also Park at column 9, lines 23-26 ("[C]ontinuity clock timer part 32 includes a plurality of timers corresponding respectively to continuity check connections in the continuity check cell channel registration part 34.")

The Examiner states in the Advisory Action that "in the claims 1, 19 and 20, it is not disclosed that the continuity cache stores an identifier that is less than the number of counters. If M is zero, then it would be substantially lower than N counters (timers)." Appellant respectfully submits that this argument appears to be directed to the limitations of dependent claim 6, rather than of independent claims 1, 19 and 20, and will hence be addressed herein. Moreover, Appellant will assume that the Examiner intended to argue that Park does disclose the limitations recited in claim 6 wherein the continuity check cache has a capacity of M entries and the set of continuity check counters includes N continuity check counters, where M is substantially less than N, rather than conceding that Park does not disclose such limitations.

As heretofore discussed above in regard to claim 1, Park teaches an arrangement wherein continuity check timer part 32 and continuity check channel registration part 34, respectively analogized by the Examiner as the recited continuity check cache and set of continuity check counters, <u>both have the same number of entries</u>. Therefore, even if the continuity check timer part 32 and continuity check channel registration part 34 could in fact be characterized as analogous to the recited continuity check cache and set of continuity check counters, Park would fail to meet the limitations of claim 6 regardless of the value assigned to M (the capacity of the continuity cache), as N (the number of continuity check counters) would have the <u>same value</u>.

It should be further noted that, if M were zero, the continuity check cache would have a capacity of zero entries. Appellant respectfully notes that claim 1, from which claim 6 depends, includes limitations wherein the controller circuitry determines if a given flow for which a protocol data unit is received in the processor has a corresponding entry in the continuity check cache, and if the given flow does not have such an entry, clearing the corresponding one of the continuity check counters and storing a flow identifier for the given flow in the continuity check cache.

Appellant respectfully submits that it would be impossible to store any flow identifiers in a continuity check cache having a capacity of zero entries. Accordingly, any arrangement in which M were zero would fail to meet the limitations of claim 1, and hence the limitations of claim 6, regardless of the value of N.

Claim 11

In addition to being allowable by virtue of its dependency from independent claim 1, dependent claim 11 recites separately patentable subject matter. Specifically, dependent claim 11 includes a limitation wherein, in conjunction with the continuity check performed for the given flow, a corresponding one of the continuity check counters is reset only a single time for a plurality of protocol data units received by the processor for the given flow within a specified time window. In an illustrative embodiment described in the specification at, for example, page , the use of a continuity check cache prevents multiple user cells on the same continuity check flow from clearing the same counter in external memory within the 0.5 second time window.

In formulating the rejection of claim 11 in the final Office Action at page 5, last paragraph, the Examiner relies on Park at column 9, lines 1-40. Appellant respectfully submits that the relied-upon portion Park fails to teach or suggest the aforementioned limitation of claim 11. Specifically, Park at column 9, lines 26-31, states that each timer is reset in response to a corresponding match signal. Park at column 9, lines 1-13, indicates that these match signals are generated responsive to a comparison of the channel identifier of each received user cell with channel identifiers of continuity check cells corresponding to active connections.

Appellant respectfully submits that, rather than disclosing the claimed arrangement, Park instead teaches an arrangement similar to that discussed in the present specification at, for example, page 8, line 22, to page 9, line 1, in which each of the cells received on a given channel will cause the timer to be reset to zero, and thus suffers from the disadvantages associated with such arrangements.

Claim 12

In addition to being allowable by virtue of its dependency from independent claim 1, dependent claim 12 recites separately patentable subject matter. Specifically, dependent claim 12 includes a limitation wherein at least one of the continuity check counters comprises a multi-bit counter with each increment of the count representing a specified time window within a designated period of time for which the continuity check is performed.

As described in the present specification at, for example, page 8, lines 3-21, an illustrative embodiment of the present invention utilizes a three-bit counter for each of the continuity check counters in order to implement a continuity check in which a timeout be reported if no cell is received for a period of 3.5 seconds +/- 0.5 seconds. A three-bit counter is therefore used in the illustrative embodiment in order to allow counting of the seven intervals, with each increment of the count corresponding to a time window having a duration of approximately 0.5 seconds. The counter is incremented for each of the 0.5 second time windows for which a user cell is not received on the corresponding flow.

In formulating the rejection of claim 12 in the final Office Action at page 6, first paragraph, the Examiner relies on Park at column 10, lines 6-10, which states that "the default value of the continuity check alarm declaration time reference register 35 may be any value other than 3.5 sec under the control of the CPU." This merely indicates that a designated period of time used in conjunction with a continuity process may be adjusted by a CPU.

Appellant respectfully submits that this portion of Park fails to teach or even suggest the limitation of dependent claim 12 wherein at least one of the continuity check counters comprises a <u>multi-bit counter with each increment of the count representing a specified time window within a designated period of time</u> for which the continuity check is performed. Rather, Park appears to teach away by teaching the use of conventional, continuously-incremented timers at, for example, column 7, line 57, to column 8, line 6, and column 9, lines 23-52.

<u>Claim 13</u>

In addition to being allowable by virtue of its dependency from independent claim 1, dependent claim 13 recites separately patentable subject matter. Specifically, dependent claim 13 includes a limitation wherein at least one of the continuity check counters comprises a three-bit counter with each increment of the count corresponding to a time window having a duration of approximately 0.5 seconds.

As described in the present specification at, for example, page 8, lines 3-21, an illustrative embodiment of the present invention utilizes a three-bit counter for each of the continuity check counters in order to implement a continuity check in which a timeout be reported if no cell is received for a period of 3.5 seconds +/- 0.5 seconds. A three-bit counter is therefore used in the illustrative embodiment in order to allow counting of the seven intervals, with each increment of the count corresponding to a time window having a duration of approximately 0.5 seconds. The counter is incremented for each of the 0.5 second time windows for which a user cell is not received on the corresponding flow.

In formulating the rejection of claim 13 in the final Office Action at page 6, second paragraph, the Examiner relies on Park at column 10, lines 6-10, which states that "the default value of the continuity check alarm declaration time reference register 35 may be any value other than 3.5 sec under the control of the CPU." This merely indicates that a designated period of time used in conjunction with a continuity process may be adjusted by a CPU.

Appellant respectfully submits that this portion of Park fails to teach or even suggest the limitation of dependent claim 13 wherein at least one of the continuity check counters comprises a three-bit counter with each increment of the count corresponding to a time window having a duration of approximately 0.5 seconds. Rather, Park appears to teach away by teaching the use of conventional, continuously-incremented timers at, for example, column 7, line 57, to column 8, line 6, and column 9, lines 23-52.

In view of the above, Appellant believes that claims 1-20 are in condition for allowance, and respectfully requests the reversal of the §103(a) rejection.

Respectfully submitted,

[illegible]

Date: August 18, 2008

Joseph B. Ryan
Attorney for Applicant(s)
Reg. No. 37,922
Ryan, Mason & Lewis, LLP
90 Forest Avenue
Locust Valley, NY 11560
(516) 759-7517

CLAIMS APPENDIX

1. A processor comprising:

controller circuitry operative to control performance of a continuity check for each of a plurality of flows of protocol data units received by the processor; and

memory circuitry comprising a continuity check cache;

wherein the continuity check cache stores an identifier for each of a subset of the plurality of flows;

wherein the controller circuitry controls access to a set of continuity check counters comprising a counter for each of the plurality of flows;

the controller circuitry determining if a given flow for which a protocol data unit is received in the processor has a corresponding entry in the continuity check cache, and if the given flow has such an entry, preventing a corresponding one of the continuity check counters from being updated, and if the given flow does not have such an entry, clearing the corresponding one of the continuity check counters and storing a flow identifier for the given flow in the continuity check cache.

2. The processor of claim 1 wherein the memory circuitry comprises an internal memory of the processor, and the continuity check cache is implemented in its entirety within the internal memory.

3. The processor of claim 1 wherein the set of continuity check counters are stored in an external memory associated with the processor.

4. The processor of claim 1 wherein at least one of the continuity checks is performed in a manner compliant with an I.610 protocol.

5. The processor of claim 1 wherein at least one of the protocol data units comprises a cell.

6. The processor of claim 1 wherein the continuity check cache has a capacity of M entries, a given one of which may correspond to the flow identifier, and the set of continuity check counters includes N continuity check counters, where M is substantially less than N.

7. The processor of claim 1 wherein one or more of the flows correspond to particular network connections.

8. The processor of claim 1 wherein each of the flows for which a flow identifier is stored in the continuity check cache has had its corresponding continuity check counter cleared upon receipt of a first protocol data unit for that flow within a specified time window.

9. The processor of claim 1 wherein each of the continuity check counters is configured so as to be incremented if one or more protocol data units are not received for the corresponding flow within a specified time window.

10. The processor of claim 1 wherein in conjunction with the continuity check performed for the given flow the continuity check fails and a timeout indication is generated if the corresponding continuity check counter reaches a particular value.

11. The processor of claim 1 wherein in conjunction with the continuity check performed for the given flow a corresponding one of the continuity check counters is reset only a single time for a plurality of protocol data units received by the processor for the given flow within a specified time window.

12. The processor of claim 1 wherein at least one of the continuity check counters comprises a multi-bit counter with each increment of the count representing a specified time window within a designated period of time for which the continuity check is performed.

13. The processor of claim 1 wherein at least one of the continuity check counters comprises a three-bit counter with each increment of the count corresponding to a time window having a duration of approximately 0.5 seconds.

14. The processor of claim 1 wherein the entries of the continuity check cache are cleared after expiration of each of a plurality of time windows for which the continuity check counters can be incremented.

15. The processor of claim 1 wherein if the continuity check cache is full when one of the plurality of flows first arrives at the processor, a particular flow identifier from the cache is removed to make room for storage of the flow identifier for the arriving flow.

16. The processor of claim 1 wherein the processor is configured to provide an interface for communication of the received protocol data units between a network and a switch fabric.

17. The processor of claim 1 wherein the processor comprises a network processor.

18. The processor of claim 1 wherein the processor is configured as an integrated circuit.

19. A method for use in a processor comprising controller circuitry operative to control performance of a continuity check for each of a plurality of flows of protocol data units received by the processor, the controller circuitry being further operative to control access to a set of continuity check counters comprising a counter for each of the plurality of flows, the method comprising the steps of:

storing an identifier for each of a subset of the plurality of flows in a continuity check cache; and

determining if a given flow for which a protocol data unit is received in the processor has a corresponding entry in the continuity check cache, and if the given flow has such an entry, preventing a corresponding one of the continuity check counters from being updated, and if the given flow does not have such an entry, clearing the corresponding one of the continuity check counters and storing a flow identifier for the given flow in the continuity check cache.

20. An article of manufacture comprising a computer readable medium having at least one computer program encoded therein for use in a processor comprising controller circuitry operative to control performance of a continuity check for each of a plurality of flows of protocol data units received by the processor, the controller circuitry being further operative to control access to a set of continuity check counters comprising a counter for each of the plurality of flows, the at least one program when executed in the processor implementing the steps of:

storing an identifier for each of a subset of the plurality of flows in a continuity check cache; and

determining if a given flow for which a protocol data unit is received in the processor has a corresponding entry in the continuity check cache, and if the given flow has such an entry, preventing a corresponding one of the continuity check counters from being updated, and if the given flow does not have such an entry, clearing the corresponding one of the continuity check counters and storing a flow identifier for the given flow in the continuity check cache.

EVIDENCE APPENDIX

None.

RELATED PROCEEDINGS APPENDIX

None.